Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Автоматизовані системи управління



Лабораторна робота № 5

**“Перевірка якості генераторів рівномірнорозподілених псевдовипадкових чисел за критеріями значимості”**

з курсу “Моделювання систем”

Виконав:

Студент гр. КН–33

Чорний Ю.І.

Перевірив:

доцент кафедри АСУ

Кузьмін О. В.

Львів 2017

**1. Мета**

Ознайомлення з тестами та критеріями значимості перевірки гіпотез та іх застосування для перевірки якості генераторів рівномірнорозподілених псевдовипадкових чисел. Об’єм роботи: 4 години.

**2. Теоретичні** **положення**

**2.1. Перевірка якості генераторів псевдовипадкових чисел**

Для визначення якості генераторів псевдовипадкових чисел використовують основні групи тестів.

1. З допомогою частотних тестів перевіряється рівномірність отриманого розподілу шляхом побудови гістограми розподілу та використання критеріїв *х*2 - квадрат або Колмогорова - Смірнова для оцінки близькості розподілу отриманої послідовності до рівномірного розподілу.
2. Тести, з допомогою яких перевіряється стохастичність (тести серій і комбінацій).
3. Кореляційні тести, з допомогою яких перевіряється незалежність елементів отриманої випадкової послідовності.

Частотні тести. Рівномірність розподілу перевіряється з допомогою гістограми, яка будується таким чином. Інтервал значень випадкової величини (0,1) розбивається на *m* підінтервалів. Теоретичне значення ймовіоності попадання випадкової величини в інтервал , де  - значення інтегральної функції розподілу. Ймовірність попадання випадкової величини в - й підінтервал для рівномірного розподілу *Рj* = 1/*m.* Для інших розподілів ймовірність можна визначити з допомогою відповідність таблиць інтегральних функцій розподілів. На основі експериментальних даних підраховуємо оцінки ймовірностей для кожного з підінтервалів: *,* де *Nj* – кількість реалізацій, що потрапили в *j*–й інтервал; *N* – загальна кількість реалізацій. Відповідність отриманого експериментального розподілу ідеальному можна визначити шляхом порівняння. На рис. 2.1 зображено гістограми емпіричного, побудованого за експериментальними даними, та ідеального рівномірного розподілу.



Рис. 2.1. Гістограма функції густини рівномірного закону розподілу, побудована за результатами експерименту

Для статистичної оцінки гіпотези про те, що сукупність емпіричних даних мало чим відрізняється від теоретичного розподілу, застосовується критерій , який запропонував Пірсон і розвинув Фішер. Статистика



де *m -* кількість інтервалів; - частота спостережень для *к*-ї групи або інтервалу; - очікувана частота для *к*-ї групи або інтервалу.

Якщо  то теоретичні частоти й частоти спостережень точно збігаються; якщо , то ці частоти збігаються не точно. При збільшенні значення  збільшується розходження між цими частотами (див. додаток). При практичному використанні статистики  висувається так звана нульова гіпотеза *Н*0, яка полягає в тому, що між теоретичним розподілом та тим, що спостерігається, немає значних розходжень. Якщо розрахункове значення ., то при заданому рівні довірчої ймовірності між розподілами наявне значне розходження і гіпотеза *Н*0 відкидається.

Правила застосування критерію :

1. Відносні значення частот або їх значення в процентах брати не можна. Необхідно використовувати результати прямих спостережень.
2. Значення частот, що спостерігаються, для кожної групи або інтервалу повинно бути не менше 5, інакше суміжні інтервали слід об’єднати.
3. Число ступенів вільності ν = *m* - *1* - *n* , де *m -* кількість інтервалів або груп; *n* - число параметрів, що визначаються шляхом досліду на основі вибіркових (експериментальних) даних для обчислення очікуваних значень частот.

Значення  можна розрахувати за формулою, зручною для проведення обчислень:



де *pk* - значення ймовірностей, з якими можна попасти в той чи інший інтервал.

Приклад. Результати роботи генератора випадкових чисел з обсягом вибірки *N* = 500 та розбиттям на 10 рівних інтервалів проміжку (0,1) (*pk* = 1/10) наведено в табл. 1.2 (для всіх інтервалів теоретичні частоти дорівнюють 50).

Таблиця 2.1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Інтервал | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Частоти спостережень | 62 | 58 | 36 | 28 | 40 | 70 | 60 | 40 | 72 | 34 |
|  | 2,88 | 1,28 | 3,92 | 9,68 | 2,00 | 8,00 | 2,00 | 2,00 | 9,68 | 5,12 |

Таким чином  = 46,56 і для рівня *Р* = 0,99 та числа ступенів вільності ν=10-1=9 значення .= 21,67 (додаток таблиця 1). Оскільки , гіпотеза Н0  відкидається.

Перевірити відповідність експериментального розподілу теоретичному можна також за допомогою критерію Колмогорова - Смірнова. Перевірку здійснюють шляхом порівняння теоретичного та експериментального інтегральних функцій розподілу (рис.2.2).



Рис.2.2. Експериментальна та теоретична інтегральні функції розподілу

Порівняння ґрунтується на вибірковій групі, в якій експериментальний розподіл має найбільше абсолютне відхилення від теоретичного. Ця різниця використовується для формування статистики

,

теоретичні значення якої при різних рівнях довірчої ймовірності наведені в додатку. Аналогічно критерію  відбувається порівняння з критичним значенням. Для прикладу результати розрахунку наведені в табл. 2.2.

Таблиця 2.2

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Інтервал | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Ймовірність | Експериментальна  інтегральна | 62  500 | 120  500 | 156  500 | 184  500 | 224  500 | 294  500 | 354  500 | 394  500 | 466  500 | 1 |
| Теоретична  інтегральна | 50  500 | 100  500 | 150  500 | 200  500 | 250  500 | 300  500 | 350  500 | 400  500 | 450  500 | 1 |
| Абсолютна різниця | | 12  500 | 20  500 | 6  500 | 16  500 | 26  500 | 6  500 | 4  500 | 6  500 | 16  500 | 0 |

Виходячи з результатів експерименту значення



i відповідно *КN* = 1,16. Гіпотезу про рівномірність генератора (згідно з даними додатку) можна прийняти при рівні довірчої ймовірності Р = 0,75, що свідчить про задовільну якість цього генератора.

У дійсності значення Δ буде дещо більшим через те, що емпірична функція розподілу F*N* (*x*) повинна будуватися з використанням 500 згенерованих реалізацій. Значення 500 реалізацій *x*1, **...** , *x*500 впорядковуються по зростанню шляхом переходу від одного значення до іншого зі співвідношенням



У результаті уточненого розрахунку внаслідок використання усереднених величин для кожного з інтервалів отримане значення буде дещоменшим за дійсне.

Наведемо кроки уточненого алгоритму для розрахунку значення К*N*:

1/ генерація значень *х*1, **...** , *х*N;

2/ сортування *хі* в порядку зростання;

3/ у кожній точці *хі* розрахунок значень 

з наступним вибором 

4**/** розрахунок значення 

Питання про застосування критеріїв χ2 та Колмогорова - Смірнова розв’язується так: при відносно малих обсягах вибірок даних (10 ≤ N ≤ 99) доцільно використовувати критерій Колмогорова -Смірнова, для великих (порядку 100 та більше) - критерій χ2 *.*

Стохастичність послідовностей псевдовипадкових чисел перевіряється за допомогою тестів комбінацій і серій. Тест комбінацій полягає у визначенні закону розподілу числа одиниць (та нулів) в *n* - розрядному двійковому числі *хі*. Закон появи *j* одиниць в *n* - розрядах двійкового числа *хі* описується виходячи з незалежності окремих розрядів біноміальним законам розподілу



де  - ймовірність виникнення одиниці в довільному розряді числа *хі*, тобто 

При фіксованій довжині вибірки, що дорівнює *N*, теоретичне число виникнення випадкових чисел з *j* одиницями в *n* – розрядах . Після розрахунку експериментальної кількості чисел з *j* одиницями в *n* - розрядах з допомогою описаних раніше критеріїв  та Колмогорова - Смірнова перевіряється гіпотеза про стохастичність.

При використанні тесту серій послідовність псевдовипадкових чисел розбивається на елементи першого *а* та другого *b* виду:



Серією називається отриманий таким способом довільний відтинок послідовності, який складається з елементів двох видів. Число елементів у серії називається довжиною серії, наприклад ***...*** *abaababb* ***...*** . Оскільки випадкові числа *a* та *b* у цій послідовності незалежні, то теоретична ймовірність того, що в серії довжиною *l* з’явиться *j* елементів *а* в *N* генераціях визначиться за формулою Бернуллі



При проведенні тесту серій оцінюються ймовірності появи серій довжиною *l* з *j* елементами *а*.

У результаті отримаємо теоретичну та експериментальну залежності *Р(j,l)*, відповідність яких перевіряється з допомогою критеріїв узгодження.

Незалежність елементів послідовності перевіряється на основі обчислення кореляційного моменту. Випадкові величини ξ та η називаються незалежними, якщо закон розподілу кожної з них не залежить від того, якого значення набула інша. Незалежність елементів послідовності {*xi*} перевіряється шляхом розгляду послідовності {*yi*}={*xi+τ*}, де *τ* - зсув послідовності. Кореляційний момент дискретних випадкових величин ξ та η з можливими значеннями *хі* та *уі* визначається за формулою



де *Рij*,- - ймовірніоть того, що (*ξ*,*η*) отримає значення (*хі ,уі* ).

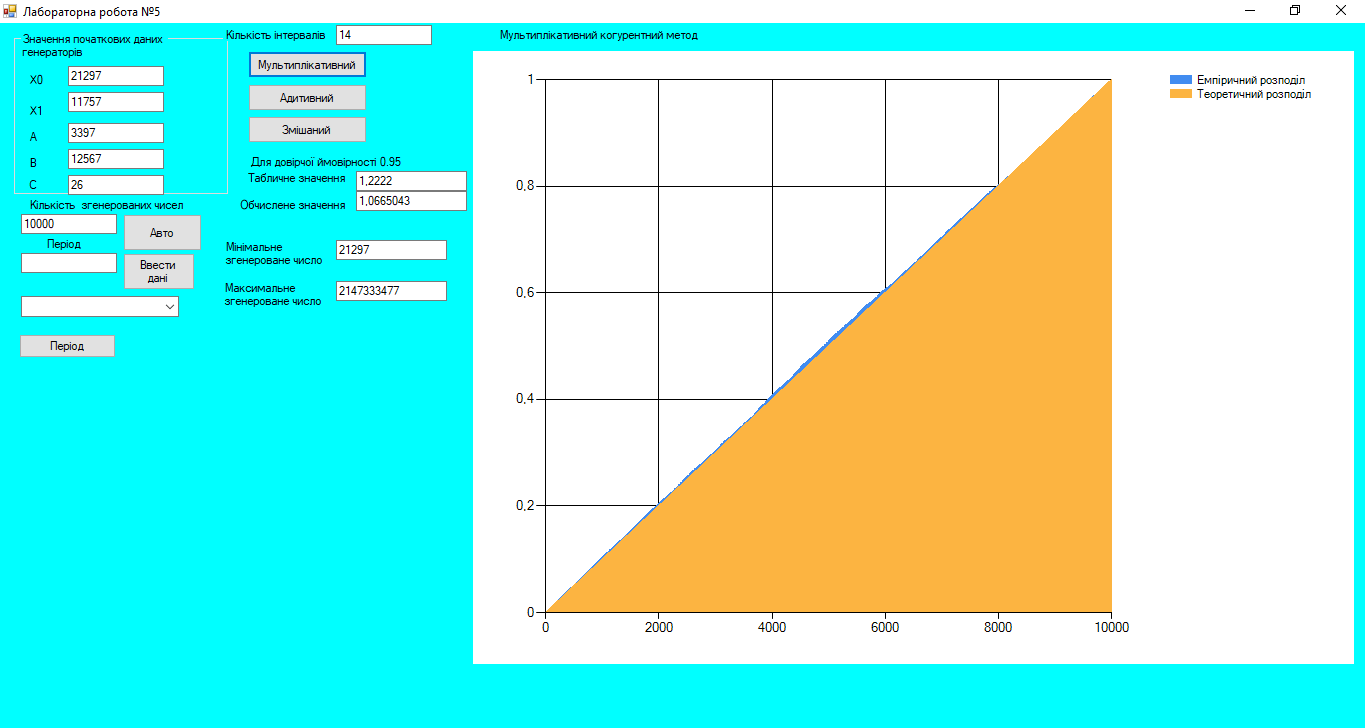
Кореляційний момент характеризує cтупінь лінійної залежності між *ξ* та *η*. Якщо *Кξη*=0, то випадкові .числа лінійно неза­лежні. Коефіцієнт кореляції *в, Рξη = Кξ η*/(*σхσу*)де *σх*, *σу -* середньоквадратичні відхилення величини відповідно *х* і *у .* Для довільного *τ* ≠ 0 *і* достатньо великих *N* з довірчою ймовірністю β повинно виконуватися співвідношення



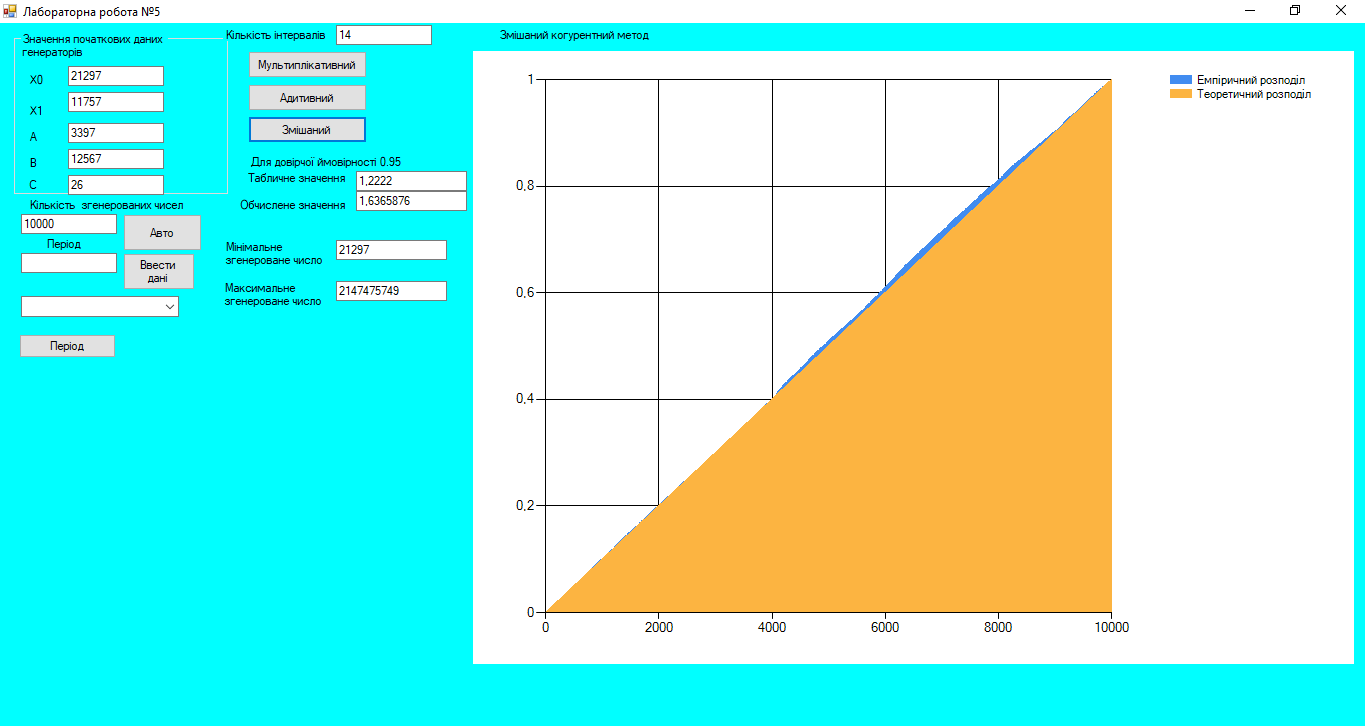
Якщо знайдене емпіричне значення *Рξη*(*τ*) знаходиться в цих межах, то з ймовірністю *β* можна стверджувати, що отримана послідовність {*xi*} задовольняє гіпотезі кореляційної незалежності.

**Скріншот програми:**

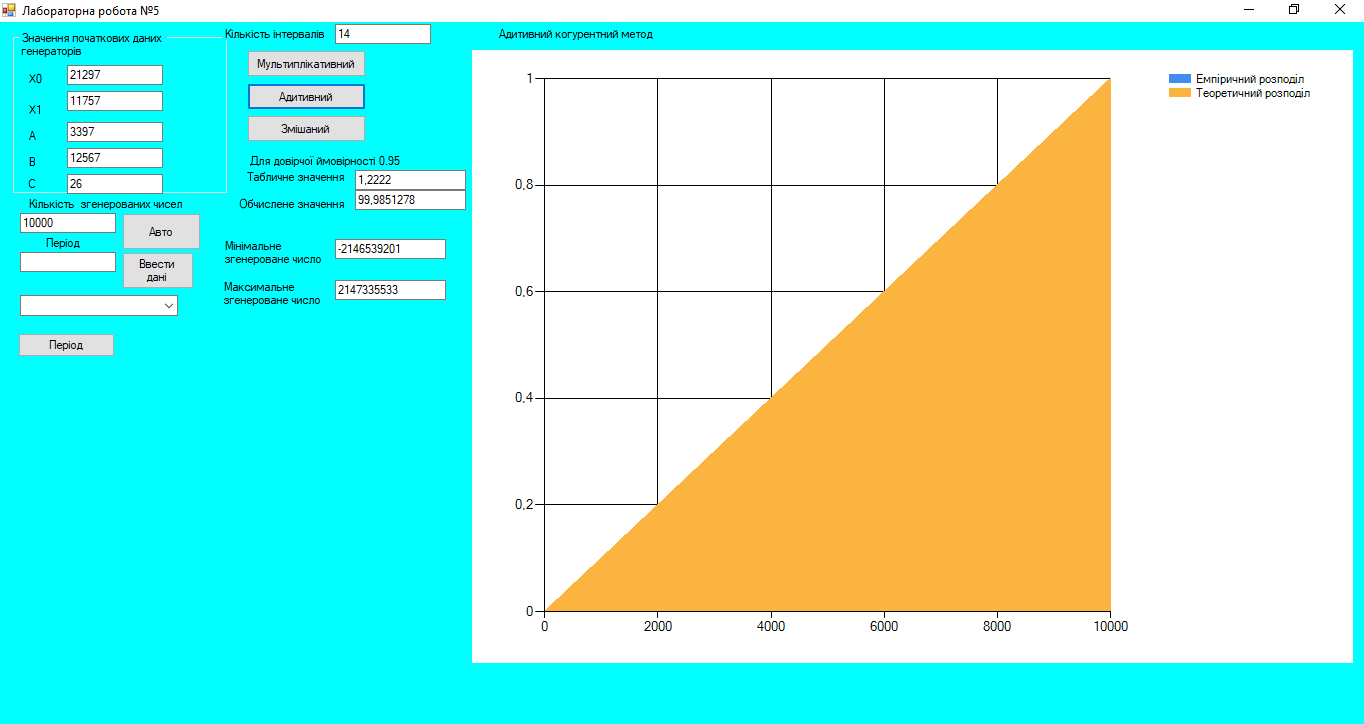
**Мультиплікативний**



**Змішаний**



**Адитивний**



**Висновок:**  Внаслідок дослідження конгруентних методів генерації псевдовипадкових чисел за рівномірним законом розподілу на ЕОМ мною було встановлено згідно гістограм розподілу, що найбільш ефективним є змішаний конгруентний розподіл.

ДОДАТОК

Код програми:

#pragma once

#include <cmath>

namespace lab5 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::Windows::Forms::DataVisualization::Charting;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::TextBox^ textBox5;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::TextBox^ textBox6;

private: System::Windows::Forms::DataVisualization::Charting::Chart^ chart1;

private: System::Windows::Forms::DataVisualization::Charting::Chart^ chart2;

private: System::Windows::Forms::DataVisualization::Charting::Chart^ chart3;

private: System::Windows::Forms::TextBox^ textBox7;

private: System::Windows::Forms::TextBox^ textBox8;

private: System::Windows::Forms::TextBox^ textBox9;

private: System::Windows::Forms::TextBox^ textBox10;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ label7;

private: System::Windows::Forms::Label^ label8;

private: System::Windows::Forms::Label^ label9;

private: System::Windows::Forms::Label^ label10;

private: System::Windows::Forms::Label^ label11;

protected:

private:

/// <summary>

/// Требуется переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Обязательный метод для поддержки конструктора - не изменяйте

/// содержимое данного метода при помощи редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::Windows::Forms::DataVisualization::Charting::ChartArea^ chartArea1 = (gcnew System::Windows::Forms::DataVisualization::Charting::ChartArea());

System::Windows::Forms::DataVisualization::Charting::Legend^ legend1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Legend());

System::Windows::Forms::DataVisualization::Charting::Series^ series1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

System::Windows::Forms::DataVisualization::Charting::Series^ series2 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

System::Windows::Forms::DataVisualization::Charting::ChartArea^ chartArea2 = (gcnew System::Windows::Forms::DataVisualization::Charting::ChartArea());

System::Windows::Forms::DataVisualization::Charting::Legend^ legend2 = (gcnew System::Windows::Forms::DataVisualization::Charting::Legend());

System::Windows::Forms::DataVisualization::Charting::Series^ series3 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

System::Windows::Forms::DataVisualization::Charting::Series^ series4 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

System::Windows::Forms::DataVisualization::Charting::ChartArea^ chartArea3 = (gcnew System::Windows::Forms::DataVisualization::Charting::ChartArea());

System::Windows::Forms::DataVisualization::Charting::Legend^ legend3 = (gcnew System::Windows::Forms::DataVisualization::Charting::Legend());

System::Windows::Forms::DataVisualization::Charting::Series^ series5 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

System::Windows::Forms::DataVisualization::Charting::Series^ series6 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

this->button1 = (gcnew System::Windows::Forms::Button());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->textBox6 = (gcnew System::Windows::Forms::TextBox());

this->chart1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Chart());

this->chart2 = (gcnew System::Windows::Forms::DataVisualization::Charting::Chart());

this->chart3 = (gcnew System::Windows::Forms::DataVisualization::Charting::Chart());

this->textBox7 = (gcnew System::Windows::Forms::TextBox());

this->textBox8 = (gcnew System::Windows::Forms::TextBox());

this->textBox9 = (gcnew System::Windows::Forms::TextBox());

this->textBox10 = (gcnew System::Windows::Forms::TextBox());

this->label6 = (gcnew System::Windows::Forms::Label());

this->label7 = (gcnew System::Windows::Forms::Label());

this->label8 = (gcnew System::Windows::Forms::Label());

this->label9 = (gcnew System::Windows::Forms::Label());

this->label10 = (gcnew System::Windows::Forms::Label());

this->label11 = (gcnew System::Windows::Forms::Label());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^ >(this->chart1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^ >(this->chart2))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^ >(this->chart3))->BeginInit();

this->SuspendLayout();

//

// button1

//

this->button1->Location = System::Drawing::Point(586, 166);

this->button1->Margin = System::Windows::Forms::Padding(2);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(84, 23);

this->button1->TabIndex = 0;

this->button1->Text = L"Згенерувати";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &Form1::button1\_Click);

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(57, 18);

this->textBox1->Margin = System::Windows::Forms::Padding(2);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(43, 20);

this->textBox1->TabIndex = 1;

this->textBox1->Text = L"12121";

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(139, 19);

this->textBox2->Margin = System::Windows::Forms::Padding(2);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(43, 20);

this->textBox2->TabIndex = 2;

this->textBox2->Text = L"25567";

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(208, 20);

this->textBox3->Margin = System::Windows::Forms::Padding(2);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(43, 20);

this->textBox3->TabIndex = 3;

this->textBox3->Text = L"12353";

//

// textBox4

//

this->textBox4->Location = System::Drawing::Point(276, 19);

this->textBox4->Margin = System::Windows::Forms::Padding(2);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(43, 20);

this->textBox4->TabIndex = 4;

this->textBox4->Text = L"21333";

//

// textBox5

//

this->textBox5->Location = System::Drawing::Point(348, 20);

this->textBox5->Margin = System::Windows::Forms::Padding(2);

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(43, 20);

this->textBox5->TabIndex = 5;

this->textBox5->Text = L"20";

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(36, 19);

this->label1->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(20, 13);

this->label1->TabIndex = 6;

this->label1->Text = L"X0";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(118, 20);

this->label2->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(20, 13);

this->label2->TabIndex = 7;

this->label2->Text = L"X1";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(187, 21);

this->label3->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(14, 13);

this->label3->TabIndex = 8;

this->label3->Text = L"A";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(255, 21);

this->label4->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(14, 13);

this->label4->TabIndex = 9;

this->label4->Text = L"C";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(327, 22);

this->label5->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(14, 13);

this->label5->TabIndex = 10;

this->label5->Text = L"B";

//

// textBox6

//

this->textBox6->Location = System::Drawing::Point(167, 47);

this->textBox6->Margin = System::Windows::Forms::Padding(2);

this->textBox6->Name = L"textBox6";

this->textBox6->Size = System::Drawing::Size(69, 20);

this->textBox6->TabIndex = 11;

this->textBox6->Text = L"10000";

//

// chart1

//

this->chart1->BackColor = System::Drawing::Color::Yellow;

chartArea1->Name = L"ChartArea1";

this->chart1->ChartAreas->Add(chartArea1);

legend1->Name = L"Legend1";

this->chart1->Legends->Add(legend1);

this->chart1->Location = System::Drawing::Point(2, 282);

this->chart1->Margin = System::Windows::Forms::Padding(2);

this->chart1->Name = L"chart1";

this->chart1->Palette = System::Windows::Forms::DataVisualization::Charting::ChartColorPalette::Bright;

series1->ChartArea = L"ChartArea1";

series1->Legend = L"Legend1";

series1->Name = L"Мультиплікативний";

series2->ChartArea = L"ChartArea1";

series2->Legend = L"Legend1";

series2->Name = L"Теоретичний";

this->chart1->Series->Add(series1);

this->chart1->Series->Add(series2);

this->chart1->Size = System::Drawing::Size(454, 203);

this->chart1->TabIndex = 12;

this->chart1->Text = L"chart1";

//

// chart2

//

this->chart2->BackColor = System::Drawing::Color::Yellow;

chartArea2->Name = L"ChartArea1";

this->chart2->ChartAreas->Add(chartArea2);

legend2->Name = L"Legend1";

this->chart2->Legends->Add(legend2);

this->chart2->Location = System::Drawing::Point(2, 88);

this->chart2->Margin = System::Windows::Forms::Padding(2);

this->chart2->Name = L"chart2";

this->chart2->Palette = System::Windows::Forms::DataVisualization::Charting::ChartColorPalette::Bright;

series3->ChartArea = L"ChartArea1";

series3->Legend = L"Legend1";

series3->Name = L"Змішаний";

series4->ChartArea = L"ChartArea1";

series4->Legend = L"Legend1";

series4->Name = L"Series2";

this->chart2->Series->Add(series3);

this->chart2->Series->Add(series4);

this->chart2->Size = System::Drawing::Size(417, 200);

this->chart2->TabIndex = 13;

this->chart2->Text = L"chart2";

this->chart2->Click += gcnew System::EventHandler(this, &Form1::chart2\_Click);

//

// chart3

//

this->chart3->BackColor = System::Drawing::Color::Yellow;

this->chart3->BackImageTransparentColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)),

static\_cast<System::Int32>(static\_cast<System::Byte>(192)), static\_cast<System::Int32>(static\_cast<System::Byte>(128)));

this->chart3->BackSecondaryColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(192)));

chartArea3->Name = L"ChartArea1";

this->chart3->ChartAreas->Add(chartArea3);

legend3->Name = L"Legend1";

this->chart3->Legends->Add(legend3);

this->chart3->Location = System::Drawing::Point(436, 241);

this->chart3->Margin = System::Windows::Forms::Padding(2);

this->chart3->Name = L"chart3";

this->chart3->Palette = System::Windows::Forms::DataVisualization::Charting::ChartColorPalette::Bright;

series5->ChartArea = L"ChartArea1";

series5->Legend = L"Legend1";

series5->Name = L"Адитивний";

series6->ChartArea = L"ChartArea1";

series6->Legend = L"Legend1";

series6->Name = L"Series2";

this->chart3->Series->Add(series5);

this->chart3->Series->Add(series6);

this->chart3->Size = System::Drawing::Size(403, 214);

this->chart3->TabIndex = 14;

this->chart3->Text = L"chart3";

//

// textBox7

//

this->textBox7->Location = System::Drawing::Point(673, 23);

this->textBox7->Margin = System::Windows::Forms::Padding(2);

this->textBox7->Name = L"textBox7";

this->textBox7->Size = System::Drawing::Size(124, 20);

this->textBox7->TabIndex = 15;

//

// textBox8

//

this->textBox8->Location = System::Drawing::Point(673, 44);

this->textBox8->Margin = System::Windows::Forms::Padding(2);

this->textBox8->Name = L"textBox8";

this->textBox8->Size = System::Drawing::Size(124, 20);

this->textBox8->TabIndex = 16;

//

// textBox9

//

this->textBox9->Location = System::Drawing::Point(673, 65);

this->textBox9->Margin = System::Windows::Forms::Padding(2);

this->textBox9->Name = L"textBox9";

this->textBox9->Size = System::Drawing::Size(124, 20);

this->textBox9->TabIndex = 17;

//

// textBox10

//

this->textBox10->Location = System::Drawing::Point(673, 104);

this->textBox10->Margin = System::Windows::Forms::Padding(2);

this->textBox10->Name = L"textBox10";

this->textBox10->Size = System::Drawing::Size(124, 20);

this->textBox10->TabIndex = 18;

//

// label6

//

this->label6->AutoSize = true;

this->label6->Location = System::Drawing::Point(78, 50);

this->label6->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(85, 13);

this->label6->TabIndex = 19;

this->label6->Text = L"Кількість чисел";

//

// label7

//

this->label7->AutoSize = true;

this->label7->Location = System::Drawing::Point(580, 6);

this->label7->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label7->Name = L"label7";

this->label7->Size = System::Drawing::Size(163, 13);

this->label7->TabIndex = 20;

this->label7->Text = L"Найбільша абсолютна різниця:";

//

// label8

//

this->label8->AutoSize = true;

this->label8->Location = System::Drawing::Point(563, 25);

this->label8->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label8->Name = L"label8";

this->label8->Size = System::Drawing::Size(105, 13);

this->label8->TabIndex = 21;

this->label8->Text = L"Aдитивного методу";

//

// label9

//

this->label9->AutoSize = true;

this->label9->Location = System::Drawing::Point(571, 47);

this->label9->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label9->Name = L"label9";

this->label9->Size = System::Drawing::Size(100, 13);

this->label9->TabIndex = 22;

this->label9->Text = L"Змішаного методу";

this->label9->Click += gcnew System::EventHandler(this, &Form1::label9\_Click);

//

// label10

//

this->label10->AutoSize = true;

this->label10->Location = System::Drawing::Point(520, 68);

this->label10->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label10->Name = L"label10";

this->label10->Size = System::Drawing::Size(149, 13);

this->label10->TabIndex = 23;

this->label10->Text = L"Мультиплікативного методу";

//

// label11

//

this->label11->AutoSize = true;

this->label11->Location = System::Drawing::Point(563, 88);

this->label11->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label11->Name = L"label11";

this->label11->Size = System::Drawing::Size(104, 39);

this->label11->TabIndex = 24;

this->label11->Text = L"Критичне значення\r\n для довірчої\r\n ймовірності 0.95\r\n";

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::Color::Yellow;

this->ClientSize = System::Drawing::Size(850, 531);

this->Controls->Add(this->label11);

this->Controls->Add(this->label10);

this->Controls->Add(this->label9);

this->Controls->Add(this->label8);

this->Controls->Add(this->label7);

this->Controls->Add(this->label6);

this->Controls->Add(this->textBox10);

this->Controls->Add(this->textBox9);

this->Controls->Add(this->textBox8);

this->Controls->Add(this->textBox7);

this->Controls->Add(this->chart3);

this->Controls->Add(this->chart2);

this->Controls->Add(this->chart1);

this->Controls->Add(this->textBox6);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->textBox5);

this->Controls->Add(this->textBox4);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button1);

this->Margin = System::Windows::Forms::Padding(2);

this->Name = L"Form1";

this->Text = L"MyForm";

this->Load += gcnew System::EventHandler(this, &Form1::Form1\_Load);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^ >(this->chart1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^ >(this->chart2))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^ >(this->chart3))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: long long multi(long long x,int a,int m)

{

return (a\*x)%m;

}

private: long long mixed(long long x,int a,int c,int m)

{

return (a\*x+c)%m;

}

private: long long additive(long long x,long long x1,long long m)

{

return (x+x1)%m;

}

private: System::Void button1\_Click(System::Object^ sender,System::EventArgs^ e)

{

int x0 = Convert::ToInt32(textBox1->Text);

int x1 = Convert::ToInt32(textBox2->Text);

int a = Convert::ToInt32(textBox3->Text);

int c = Convert::ToInt32(textBox4->Text);

int b = Convert::ToInt32(textBox5->Text);

long long int m = pow(2,b);

int n=Convert::ToInt32(textBox6->Text);

this->chart1->Series["Мультиплікативний"]->Points->Clear();

this->chart2->Series["Змішаний"]->Points->Clear();

this->chart3->Series["Адитивний"]->Points->Clear();

this->chart1->Series["Теоретичний"]->Points->Clear();

this->chart2->Series["Series2"]->Points->Clear();

this->chart3->Series["Series2"]->Points->Clear();

long long x\_t1,x\_t2;

int k=1+3.2\*log10(n)+0.99;

int max=1,min=0; double len = 1.0/k;

int \*n\_ = new int[k];

x\_t1 = additive(x0,x1,m);

x\_t2 = additive(x1,x\_t1,m);

double j=(double)x\_t1/m;

for(int i=0; i<k; i++)

n\_[i]=0;

for(int i=0; i<n; i++){

if(i%2==0){

x\_t1 = additive(x\_t1,x\_t2,m);

j=(double)x\_t1/m;

for(int i=0; i<k; i++)

if(j >= i\*len && j < i\*len+len){

n\_[i]++; break;

}

}

else{

x\_t2 = additive(x\_t1,x\_t2,m);

j=(double)x\_t2/m;

for(int i=0; i<k; i++)

if(j >= i\*len && j < i\*len+len){

n\_[i]++; break;

}

}

}

double \*del = new double[k];

double kp=0,kp1=0;

for(int i=0; i<k; i++){

kp+=(double)n\_[i]/n;

kp1+=1.0/k;

del[i]=abs(kp1-kp);

this->chart3->Series["Адитивний"]->Points->AddXY(i+1,kp);

this->chart3->Series["Series2"]->Points->AddXY(i+1,kp1);

}

double d\_kr=1.36/sqrt(n);

textBox10->Text=d\_kr.ToString();

double max\_d=del[0];

for(int i=1; i<k; i++)

if(del[i]>max\_d) max\_d=del[i];

double k\_n1=max\_d;

textBox7->Text=k\_n1.ToString();

x\_t1=mixed(x0,a,c,m);

for(int i=0; i<k; i++)

n\_[i]=0;

j=(double)x\_t1/m;

for(int i=0; i<k; i++)

if(j >= i\*len && j < i\*len+len) n\_[i]++;

for(int i=0; i<n; i++){

x\_t1 = mixed(x\_t1,a,c,m);

j=(double)x\_t1/m;

for(int i=0; i<k; i++)

if(j >= i\*len && j < i\*len+len){

n\_[i]++; break;

}

}

kp=0,kp1=0;

for(int i=0; i<k; i++){

kp+=(double)n\_[i]/n;

kp1+=1.0/k;

del[i]=(kp1-kp);

this->chart2->Series["Змішаний"]->Points->AddXY(i+1,kp);

this->chart2->Series["Series2"]->Points->AddXY(i+1,kp1);

}

max\_d=del[0];

for(int i=1; i<k; i++)

if(del[i]>max\_d) max\_d=del[i];

textBox8->Text=max\_d.ToString();

x\_t1=multi(x0,a,m);

for(int i=0; i<k; i++)

n\_[i]=0;

j=(double)x\_t1/m;

for(int i=0; i<k; i++)

if(j >= i\*len && j < i\*len+len) n\_[i]++;

for(int i=0; i<n; i++){

x\_t1 = multi(x\_t1,a,m);

j=(double)x\_t1/m;

for(int i=0; i<k; i++)

if(j >= i\*len && j < i\*len+len){

n\_[i]++; break;

}

}

kp=0,kp1=0;

for(int i=0; i<k; i++){

kp+=(double)n\_[i]/n;

kp1+=1.0/k;

del[i]=(kp1-kp);

this->chart1->Series["Мультиплікативний"]->Points->AddXY(i+1,kp);

this->chart1->Series["Теоретичний"]->Points->AddXY(i+1,kp1);

}

max\_d=del[0];

for(int i=1; i<k; i++)

if(del[i]>max\_d) max\_d=del[i];

textBox9->Text=max\_d.ToString();

}

private: System::Void label9\_Click(System::Object^ sender,System::EventArgs^ e)

{

}

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void chart2\_Click(System::Object^ sender, System::EventArgs^ e) {

}

};

}